CSE 250 Recitation

11/27-11/28 : Hash Tables

PA3 Implementation Tips

In PA3 you will be de-anonymizing data based on a person's voter record and health record. Each record contains a birthday and a zip code field, which will be used to determine unique matches.

How can we tell if two records are "unique"?

How do we deal with null values? (they act as wildcards)

PA3 Implementation Tips

The four functions you are being asked to implement: **`loadHealthRecords()`**, **`loadVoterRecords()`**, **`identifyPersons()`**, **`computeHealthRecordDist()`**.

`LoadHealthRecord()/LoadVoterRecord()`

- This is where you want to start working on PA3. You are tasked with properly reading data from a csv file and storing said data into a health or voter record.
- While these functions seem very similar, the csv files have different formats and improperly stored data that must be accounted for.

PA3 Implementation Tips

`identifyPersons()`

- This is where you are being tasked with taking the outputs of the load functions and using them to match a person's name from the voter records with their health record.
- A **safe way to start to is find a way to find exact, unique matches** and work from there to figure out how to handle null values.
- While you write this function, don't be afraid to add hash tables with keys other than a BZPair.

PA3 Implementation Tips

`computeHealthRecordDist()`

- This function has two different outputs, depending on the attribute provided as an argument.
- The blood type attribute asks you to map a blood type to the percentage of the population that has that blood type.
- The allergy attribute asks you to map the different allergies to the percentage of the population that has that allergy.

Hashing

Take the items A-E and their corresponding hash values:

- hash(A) = 636
- hash(B) = 712
- hash(C) = 459
- hash(D) = 12
- hash(E) = 154

1. Start with a 5-bucket hash table (with chaining) and insert the above items
2. Rehash the table, doubling its size to 10

Open Addressing

Take the items A-E and their corresponding hash values:

- hash(A) = 636
- hash(B) = 712
- hash(C) = 459
- hash(D) = 12
- hash(E) = 154

1. Start with a 5-bucket hash table (with open addressing) and insert the above items
2. Run through the process of looking up records A-E and F (hash(F) = 232)
3. Remove item B
4. Rehash, doubling the array size to 10 and repeat steps 2 and 3

Cuckoo Hashing

Take the items A-E and their corresponding hash values:

- $hash_1(A) = 312$ $hash_2(A) = 636$
- $hash_1(B) = 242$ $hash_2(B) = 712$
- $hash_1(C) = 684$ $hash_2(C) = 459$
- $hash_1(D) = 871$ $hash_2(D) = 12$
- $hash_1(E) = 154$ $hash_2(E) = 939$

1. Start with a 5-bucket hash table (with cuckoo hashing) and insert the above items (rehash as needed)